REMARKS

The examiner has objected to the "numbering" of the steps in claim 43. Claim 43 has been appropriately amended.

Claim 31 stands rejected under 35 U.S.C. 112, second paragraph as being indefinite. Applicant has amended claim 31 and submits that amended claim 31 complies with the requirements of 35 U.S.C. 112, second paragraph.

Claims 27-28, 32, 35-38, 40, 42-44, 46, and 48-49 stand rejected under 35 U.S.C. 102(e) as being anticipated by Altberg.

The present invention, as defined by claim 27, relates to a computer readable medium having an executable application recorded thereon. The executable application includes a program, one or more encrypted sub-routines, and a decryption routine. The program is executed in response to execution of the executable application by a computer system. During execution, the program requires access to the sub-routines and the decryption routine is operable to detect whether a required sub-routine is already available within the computer system. If a sub-routine is already available, the decryption routine causes the program to use the sub-routine. If the sub-routine is not already available, the decryption routine decrypts the required encrypted sub-routine into an executable form, at least when access to the sub-routine is required by the program. Applicant observes that the executable application, as defined by claim 27, is necessarily self-contained. That is, all software components necessary for the application's execution are present in the application.

Altberg describes a computer system 200 that includes, among other things, an application 205, a shared library DLL 210, a plurality of DLL files 215, and an installer module 220 (e.g. Figure 2 of Altberg). The application requires one or more DLL files in order to execute properly (e.g. col. 5, lines 21-23 of Altberg). Upon executing the application, the application initially calls the shared library DLL. The shared library DLL then checks whether the DLL files required by the application are already installed on the computer, i.e. whether all required DLL files are present at 215 (e.g. col. 5, lines 49-60 of Altberg). If a file required by the application is missing, the application then calls the installer module (col. 5, lines 66-67 of Altberg). The installer module comprises at least all DLL files required by the application. Upon being launched, the installer module determines whether there is sufficient space on the hard disk of the computer system to install the missing required files (e.g. col. 6, lines 27-29 of Altberg). If sufficient space is available, the installer module installs the missing files to the predetermined location at which all other DLL files 215 are located. Once the missing files have been installed, the application execution is completed (col. 7, lines 32-35 of Altberg).

With reference to claim 27 of the present application, the examiner equates the application 205 of Altberg to claim 27's executable application and to claim 27's program, which comprises part of the executable application. Additionally, the shared library DLL 210, which determines whether a required sub-routine is already available, and the installer module 220, which installs missing sub-routines, are equated to the decryption routine of claim 27. However, Altberg's FIG. 2, referenced by the examiner, shows the application 205, the shared library DLL 210 and the installer module 220 as separate and distinct executable programs. There is no suggestion in Altberg of providing an executable application that includes the program, the shared library DLL and the installer module in a single, self-contained executable application.

In contrast, as described above, the present invention is directed to a single executable application that contains a program, one or more encrypted sub-routines, and a decryption routine. When the application is executed, the program is executed. During execution, the program requires access to one or more sub-routines. When access to a sub-routine is required, the decryption routine detects whether the sub-routine is already available within the computer system. If the sub-routine is not available, the decryption routine decrypts the required encrypted sub-routine for access by the program.

Not only does Altberg fail to teach the executable application defined by claim 27, it would also not be obvious to modify the teachings of Altberg so as to provide the application 205, shared library DLL 210 and the installer module 220 as a single executable. It is an essential feature of the computer system described by Altberg that the shared library DLL 210 and the DLL files 215 (i.e. the required sub-routines) are shared by other applications to be executed by the computer system (e.g. col. 5, lines 23-28 of Altberg). To this end, the DLL files 215 are stored at predetermined locations (e.g.

col. 5, lines 36-48 of Altberg). Additionally, the installer module 220 stores not only DLL files required by the application 205, but also DLL files that may be required by other applications. It would not therefore be obvious to include the shared library DLL 210 and/or the installer module 220 as part of the application 205 to be executed.

In view of the foregoing, applicant submits that Altberg does not disclose every element of the invention defined by claim 27 as is required by 35 U.S.C. 102(e). Applicant therefore submits that claim 27 is patentable over Altberg. It follows that dependent claims 28, 29 and 31-36 are also patentable.

Applicant further submits the above arguments apply equally to independent claims 37, 43 and 49. Applicant therefore submits that claims 37, 43 and 49 is patentable over Altberg. It follows that dependent claims 38-42, 44 and 46-48 and 50 are also patentable.

Respectfully submitted,

[illegible]

John Smith-Hill
Reg. No. 27,730

SMITH-HILL & BEDELL, P.C.
16100 N.W. Cornell Road, Suite 220
Beaverton, Oregon 97006

Tel. (503) 574-3100
Fax (503) 574-3197

Docket: FORR 2275